Curs 4

Analiza cerinţelor

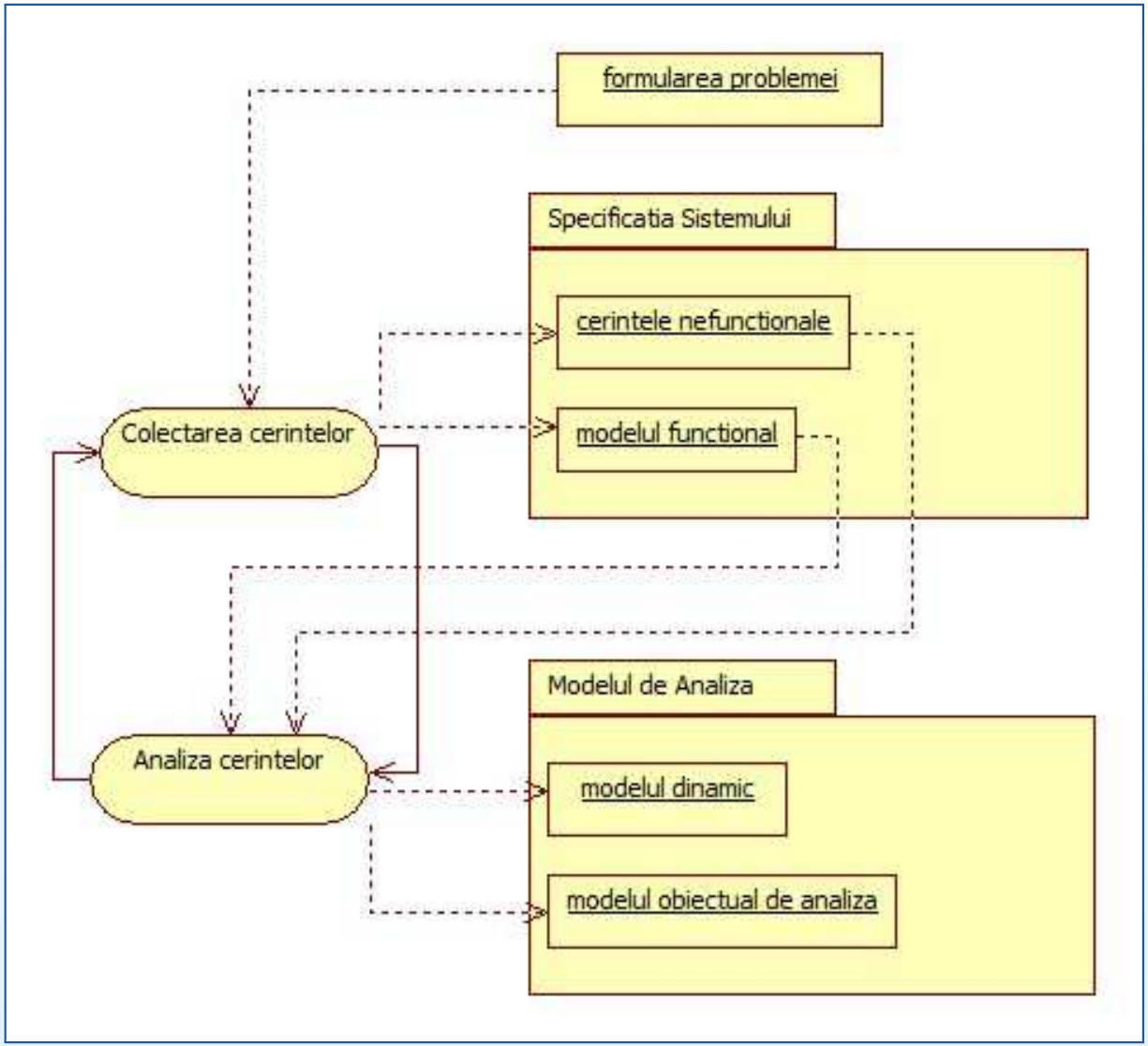
Curs bazat pe ***B. Bruegge and A.H. Dutoit***

"Object-Oriented Software Engineering using UML, Patterns, and Java"

Sumar Curs 4

- Analiza cerinţelor - obiective
- Analiza cerinţelor - concepte
- Analiza cerinţelor - activităţi tehnice
- Documentul de analiză a cerinţelor

Ingineria cerinţelor - activităţi şi modele

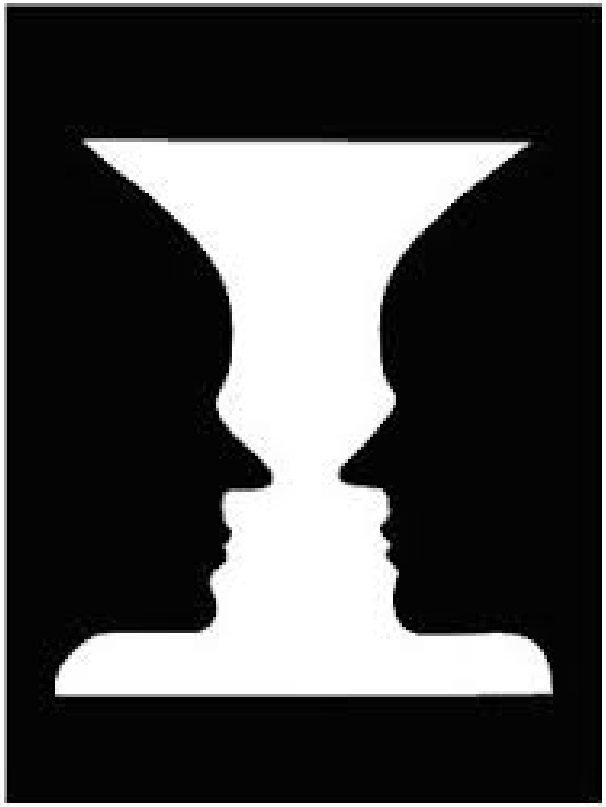


Analiza cerinţelor - obiective

- Scopul analizei cerinţelor este realizarea unui model al sistemului (*model de analiză*, eng. *analysis model*) corect, complet, consistent şi neambiguu
- Accentul este pus pe structurarea şi formalizarea cerinţelor colectate în etapa anterioară
- Formalizarea permite identificarea de ambiguităţi, inconsistenţe şi incompletitudini în descrierea cerinţelor, ce sunt adresate prin discuţii cu clientul/utilizatorii şi modificarea specificaţiei sistemului
- Colectarea şi analiza cerinţelor sunt activităţi concurente, ce se desfăşoară iterativ-incremental

Imagini multistabile

- Ce reprezintă?
- Ambiguitate - daca ar fi o specificare, ce model am construi?



Analiza cerinţelor - concepte

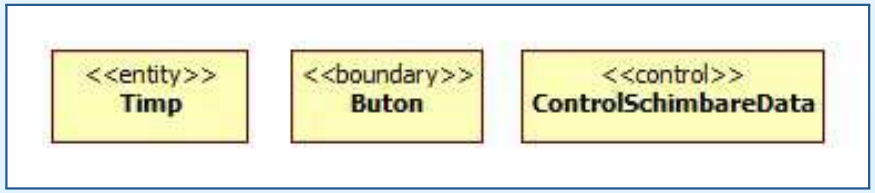
- *Modele obiectuale* şi *modele dinamice*
- Clase *entity*, *boundary* şi *control*
- *Generalizare / specializare*

Modele obiectuale şi modele dinamice

- *Modelul obiectual de analiză*
  - Surprinde conceptele manipulate de sistem, proprietăţile şi relaţiile acestora
  - Se reprezintă cu ajutorul *diagramei de clase*
  - O clasă din modelul de analiză reprezintă o abstractizare pentru una sau mai multe clase din codul sursă (care, în general, vor conţine şi un număr mult mai mare de atribute şi asocieri)
- *Modelul dinamic*
  - Surprinde comportamentul sistemului
  - Reprezentat cu ajutorul *diagramelor de secvenţă* şi al *diagramelor de tranziţie a stărilor*
    - Diagramele de secvenţă surprind interacţiunile dintr-o mulţime de obiecte în cadrul unui caz de utilizare
    - Diagramele de tranziţie a stărilor reprezintă comportamentul unui singur obiect (sau grup de obiecte strâns cuplate)
  - Modelul dinamic permite identificarea responsabilităţilor claselor individuale, precum şi identificarea de clase / atribute / asocieri noi
- Modelele de analiză surprind doar concepte / atribute / relaţii / comportamente percepute de utilizatori

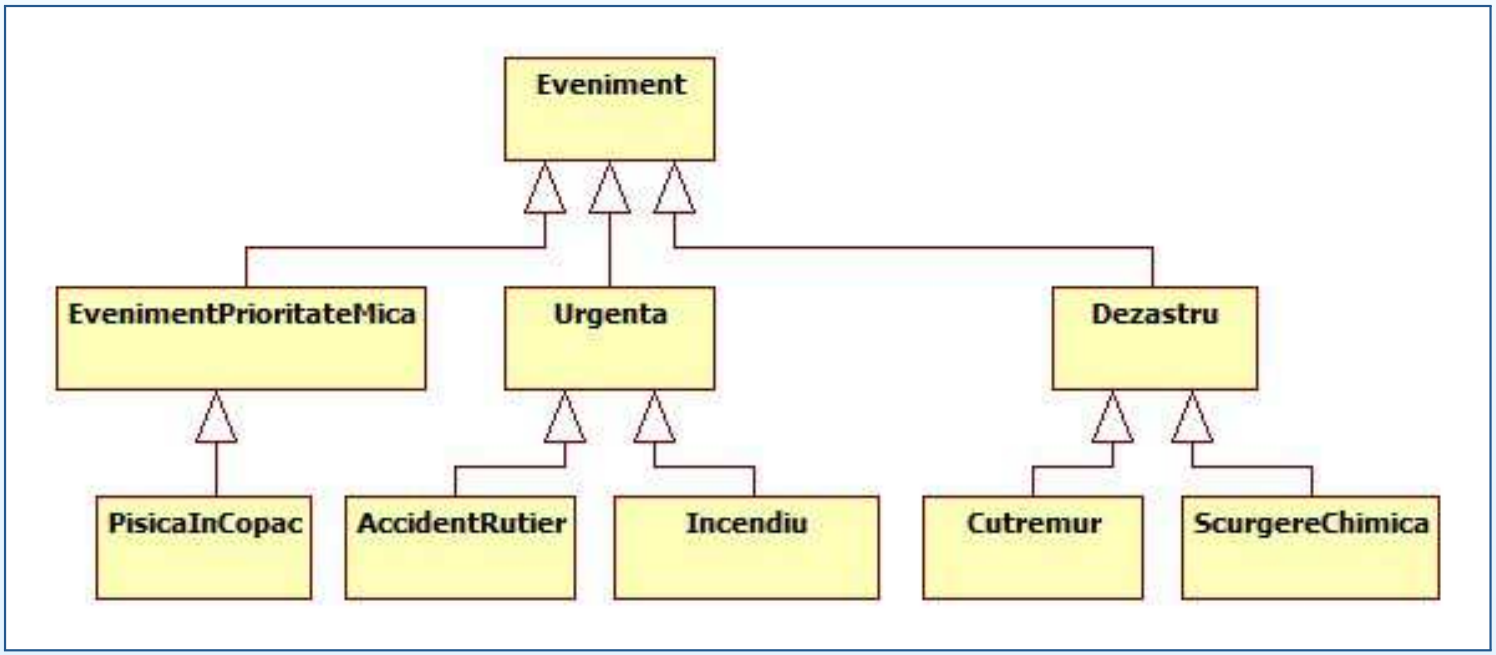
Clase *entity*, *boundary* şi *control*

- Modelul obiectual de analiză constă din clase *entity*, *boundary* şi *control*
  - Clase *entity* - responsabile de informaţia persistentă din sistem
    - Ex.: clasa *Timp*, clasa *Eveniment*
  - Clase *boundary* - responsabile de interacţiunea actorilor cu sistemul
    - Ex.: clasa *Buton*, clasa *FormăRaportareUrgenţă*
  - Clase *control* - responsabile de realizarea cazurilor de utilizare
    - Ex.: clasa *ControlSchimbareData*, clasa *ControlGestionareUrgenţă*
- Avantaje ale acestui tip de clasificare
  - Clase / obiecte mai mici, specializate
  - Modele mai uşor de modificat
    - Interfaţa (obiectele *boundary*) are o probabilitate mai mare de a fi modificată decât funţionalitatea de bază (obiectele *entity* şi *control*)
- Notaţie: stereotipuri UML + convenţii de denumire



Generalizare / specializare

- Permit organizarea conceptelor în ierarhii
  - *Generalizare* = activitate de modelare ce are drept scop identificarea unor concepte abstracte, pornind de la cele de nivel jos
  - *Specializare* = activitate de modelare ce are drept scop identificarea unor concepte specializate, pornind de la cele de nivel înalt



Analiza cerinţelor - activităţi

- Identificarea claselor *entity*
- Identificarea claselor *boundary*
- Identificarea claselor *control*
- Maparea cazurilor de utilizare la obiecte cu diagrame de secvenţă
- Identificarea asocierilor
- Identificarea agregărilor
- Identificarea atributelor
- Modelarea comportamentului obiectelor cu diagrame de stări
- Identificarea ierarhiilor de clase
- Revizuirea modelului de analiză

  - Activităţile sunt ghidate de euristici
  - Calitatea produselor lor depinde de experienţa dezvoltatorilor în aplicarea euristicilor şi metodelor aferente

Cazul de utilizare (CU) *Raportează Urgenţă* a SGA

Nume	*Raportează Urgenţă*
Participanţi	Iniţiat de *OfiţerTeren* Comunică cu *Dispecer*ul
Flux de evenimente	1. *Ofiţerul* activează funcţia *Raportează urgenţă* a terminalului său. 2. Sistemul *SGA* afişează un formular *Ofiţerului*. Formularul include componente privind tipul urgenţei (general, incendiu, accident auto), nivelul de alertă, locaţia, descrierea şi resursele solicitate. 3. *Ofiţerul* completează formularul, inserând cel puţin tipul urgenţei şi descrierea situaţiei. El poate descrie şi soluţii viabile la situaţia de urgenţă şi poate solicita resurse specifice. După completare, *Ofiţerul* trimite formularul apăsând butonul *Trimite raport*. 4. Sistemul primeşte formularul şi notifică *Dispecerul*. 5. *Dispecerul* verifică informaţia primită şi creează un nou *Eveniment* în baza de date prin invocarea cazului de utilizare *DeschideCazNou*. Toate informaţiile din formularul primit sunt asociate automat evenimentului creat. *Dispecerul* alege un răspuns prin alocarea de resurse la eveniment (prin cazul de utilizare *AlocareResurse*) şi confirmă primirea formularului printr-un mesaj către ofiţer. 6. *Ofiţerul* primeşte confirmarea şi răspunsul ales.
...	

Identificarea claselor *entity*

- Procesul de identificare porneşte de la descrierea cazurilor de utilizare
- *Analiza limbajului natural* (eng. *natural language analysis* [Abbot 1983]) oferă un set util de euristici pentru identificarea claselor, obiectelor, atributelor, operaţiilor, relaţiilor şi constrângerilor
  - Părţilor de vorbire le corespund elemente din model

Parte de vorbire	**Componentă din model**	**Exemple**
Substantiv propriu	Obiect	Alice
Substantiv comun	Clasă sau atribut	Ofiţer din teren
		Descrierea evenimentului
Verb "a face"	Operaţie sau asociere	Creează, trimite
Verb "a fi"	Moştenire	Este un/o
Verb "a avea"	Agregare	Are, constă din
Verb modal	Constrângere	Trebuie să fie

Identificarea claselor *entity* (cont.)

- Avantaje
  - Metodă focusată pe terminologia utilizatorilor
  - Rezultate bune atunci când se doreşte identificarea claselor candidat pe baza unor descrieri scurte (fluxul unui scenariu sau caz de utilizare)
- Dezavantaje
  - Calitatea modelului obiectual e dependentă de stilul de specificare al analistului (claritate, consecvenţă în utilizarea termenilor)
  - Numărul substantivelor e mai mare decât cel al claselor (unele substantive sunt sinonime sau sunt atribute ale altor clase)

- Euristici suplimentare de utilizat pentru identificarea claselor *entity*
  - Termeni pe care dezvoltatorii şi utilizatorii trebuie să îi clarifice pentru a înţelege cazul de utilizare
  - Substantive care se repetă în descrierea cazului de utilizare (ex: *Eveniment*)
  - Entităţi din lumea reală pe care sistemul trebuie să le gestioneze (ex.: *Dispecer*, *Resursă*)
  - Activităţi din lumea reală pe care sistemul trebuie să le gestioneze (ex.: *PlanDeOperaţiiUrgenţă*)

CU *Raportează*Urgenţă - clase *entity*

Nume clasă	Descriere
OfiţerTeren	Ofiţer de poliţie sau pompieri la datorie. Un *OfiţerTeren* este identificat prin număr de ecuson şi nu poate fi alocat la două *Eveniment*e simultan.
RaportUrgenţă	Raport iniţial despre un *Eveniment*, trimis de un *OfiţerTeren* unui *Dispecer*. Primirea unui *RaportUrgenţă* determină, de obicei, crearea unui *Eveniment* de către *Dispecer*. Un *RaportUrgenţă* conţine un nivel de urgenţă, un tip (incendiu, accident rutier, etc.), o locaţie şi o descriere.
Dispecer	Ofiţer de poliţie care gestionează *Eveniment*ele. Un *Dispecer* deschide, documentează şi închide un *Eveniment*, ca şi răspuns al unui *RaportUrgenţă* sau a altui tip de comunicare cu un *OfiţerTeren*. Un *Dispecer* este identificat prin număr de ecuson.
Eveniment	Situaţie care necesită intervenţia unui *OfiţerTeren*. Un *Eveniment* poate fi raportat în sistem de către un *OfiţerTeren*, sau de către orice altă entitate externă. Un *Eveniment* constă dintr-o descriere, un răspuns, o stare (deschis, închis, documentat), o locaţie şi un număr de *Ofiţer*i*Teren*.

Identificarea claselor *boundary*

- Clasele *boundary* reprezintă interfaţa sistemului cu actorii
  - În cadrul fiecărui caz de utilizare, fiecare actor interacţionează cu cel puţin un obiect de tip *boundary*
  - Obiectele *boundary* colectează inputul actorilor şi îl transformă într-o formă utilizabilă de către obiectele *entity* şi *control*
- Euristici de determinare a claselor *boundary*
  - Identifică controalele de care utilizatorii au nevoie pentru a iniţia un caz de utilizare (ex.: *ButonRaportareUrgenţă*)
  - Identifică formele de care utilizatorii au nevoie pentru a introduce date în sistem (ex.: *FormaRaportUrgenţă*)
  - Identifică notificările şi mesajele folosite de sistem pentru a răspunde actorilor (ex.: *NotificareConfirmare*)
  - În situaţia în care un caz de utilizare implică mai mulţi actori, identifică terminalele acestora pentru a referi interfaţa utilizator aferentă (ex.: *StatiaDispecer*)
  - Clasele *boundary* reprezintă elemente de interfaţă de granularitate mare
  - Elementele de interfaţă trebuie descrise folosind exclusiv vocabularul utilizatorilor

CU *Raportează Urgenţă* - clase *boundary*

Nume clasă	Descriere
ButonRaportareUrgenţă	Buton utilizat de *OfiţerTeren* pentru a iniţia cazul de utilizare aferent.
FormăRaportUrgenţă	Formă deschisă unui *OfiţerTeren* pe o *StaţieOfiţerTeren* în momentul selectării funcţiei *Raportează Urgenţă*. Forma conţine câmpuri aferente tuturor atributelor unui raport de urgenţă şi un buton pentru a fi trimisă.
FormăEveniment	Formă utlizată pentru crearea unui *Eveniment*. Este prezentată unui *Dispecer* pe o *StaţieDispecer* în momentul primirii unui *RaportUrgenţă*. *Dispecer*ul o utilizează şi pentru alocarea resurselor şi trimiterea confirmării către *OfiţerTeren*.
NotificareConfirmare	Notificare utilizată pentru a afişa confirmarea *Dispecer*ului către *Ofiţer*ul*Teren*
StaţieDispecer	Calculator utilizat de către *Dispecer*
StaţieOfiţerTeren	Calculator portabil utilizat de către *OfiţerTeren*

Identificarea claselor *control*

- Obiectele *control* sunt responsabile de coordonarea obiectelor *boundary* şi *entity*
  - Obiectele *control* nu au, de obicei, un corespondent în lumea reală
  - Un obiect *control* eate asociat, de regulă, unui caz e utilizare: este creat la începutul cazului de utilizare şi distrus la finalizarea acestuia.
  - Ex.: Obiectele *control* gestionează comportamentul legat de succesiunea formelor, cozi undo şi istoric, transmiterea informaţiei într-un mediu distribuit, etc.
- Euristici pentru determinarea obiectelor *control*
  - Identifică un control per caz de utilizare
  - Identifică un control per actor într-un caz de utilizare
  - Durata de viaţă a obiectului *control* corespunde duratei de execuţie a cazului de utilizare sau duratei unei sesiuni utilizator

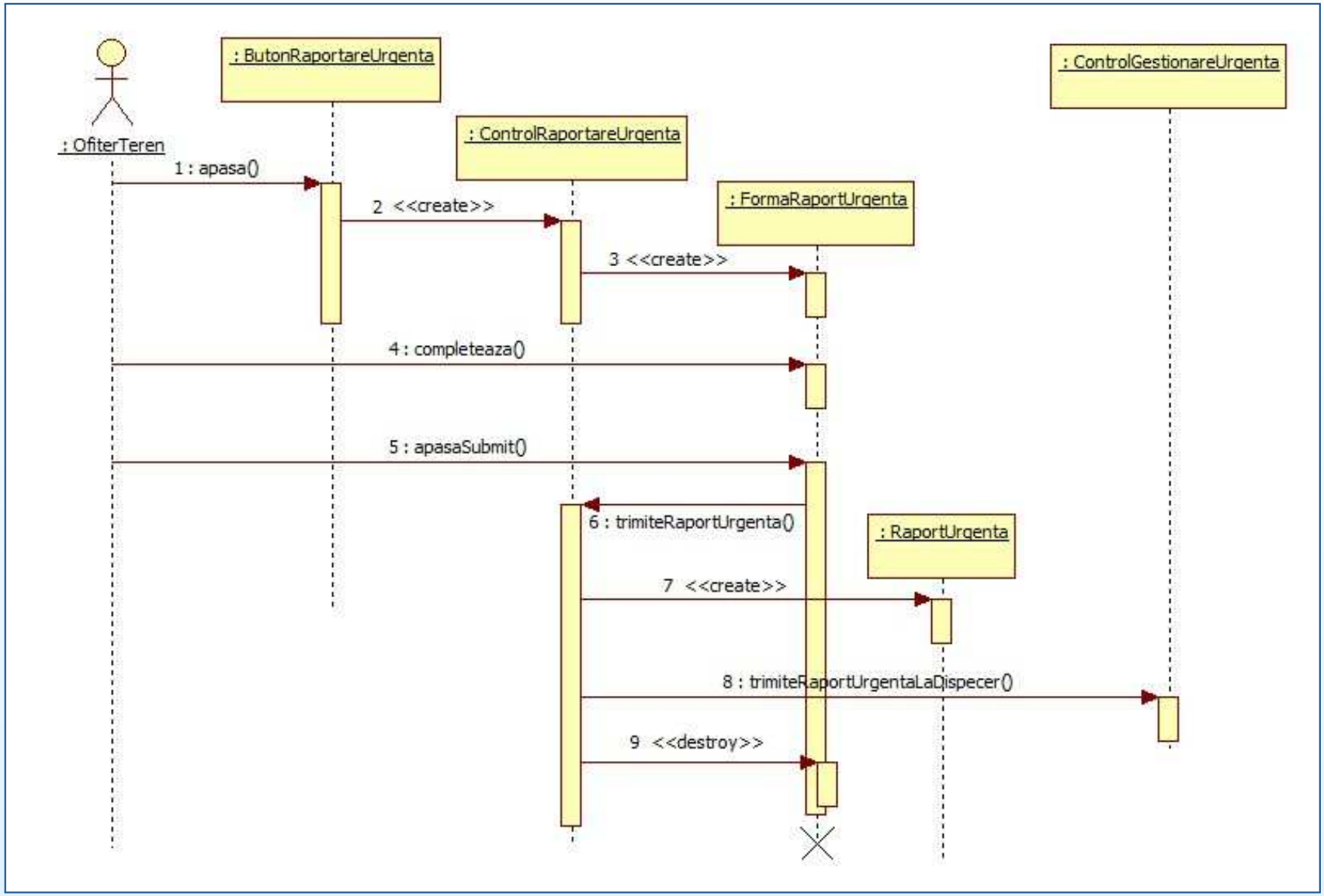
CU *Raportează Urgenţă* - clase *control*

Nume clasă	Descriere
ControlRaportareUrgenţă	Gestionează raportarea unei urgenţe de pe o *StaţieOfiţerTeren*. Un obiect de acest tip este creat în momentul în care un *OfiţerTeren* apasă butonul "Raportează Urgenţă". Ulterior, acesta creează un obiect de tip *FormăRaportUrgenţă*, pe care o afişează ofiţerului. La trimiterea formularului, acest obiect colectează informaţia completată, creează un obiect de tip *RaportUrgenţă* şi îl trimite *Dispecer*ului. Obiectul control aşteaptă apoi o confirmare de la staţia dispecerului. În momentul primirii confirmării, creează un obiect de tip *NotificareConfirmare* şi o afişează ofiţerului.
ControlGestionareUrgenţă	Gestionează raportarea unei urgenţe pe o *StaţieDispecer*. Un obiect de acest tip este creat în momentul primirii unui *RaportUrgenţă*. Ulterior, acesta creează o *FormăEveniment* şi o afişează *Dispecer*ului. După ce dispecerul creează un *Eveniment*, alocă resurse şi trimite o confirmare, obiectul control forwardează confirmarea *Ofiţer*ului*Teren*.

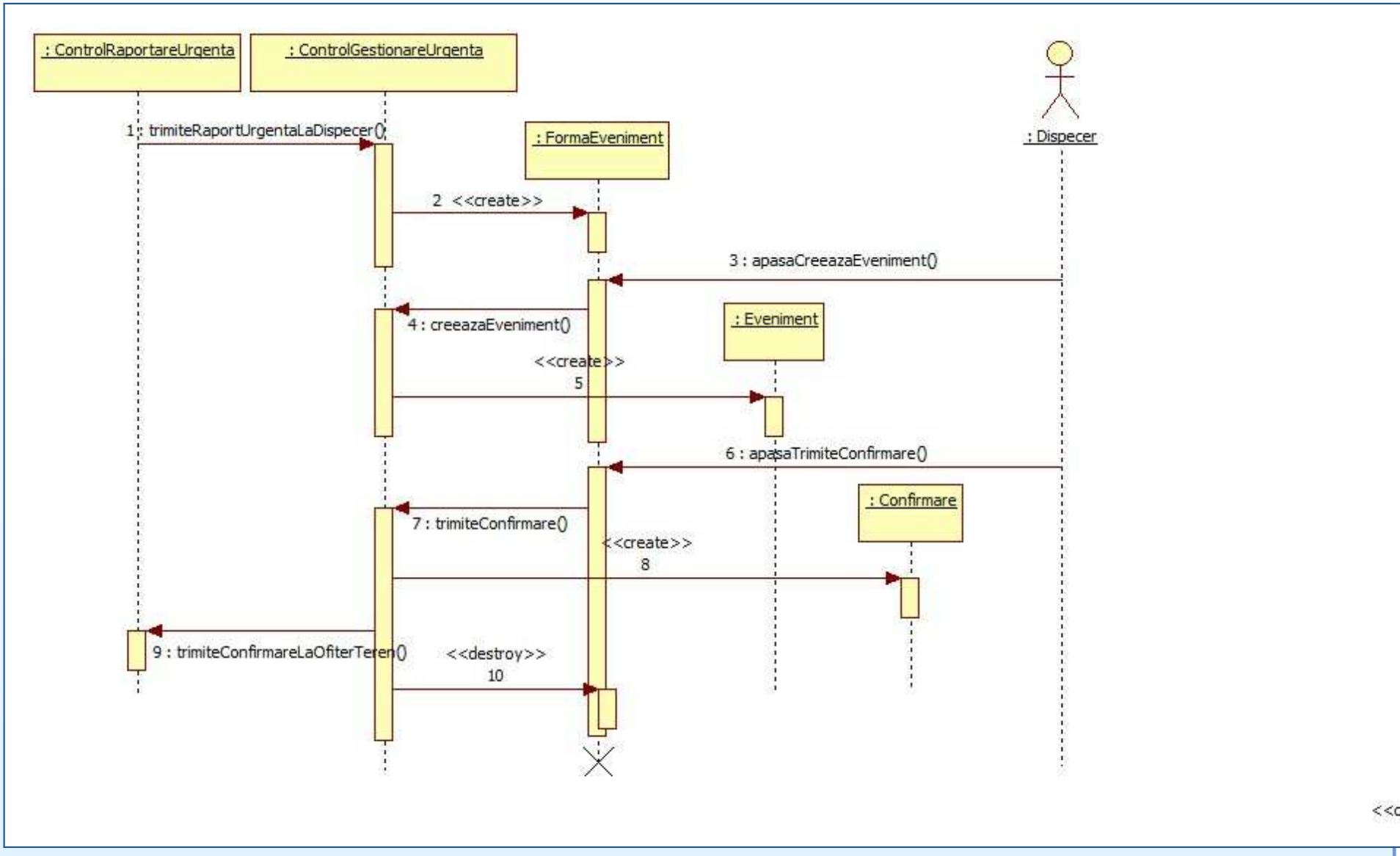
Realizarea diagramelor de secvenţă

- Diagramele de secvenţă realizează legătura dintre clase/obiecte şi scenarii/cazuri de utilizare
  - Prezintă modul în care comportamentul descris de un scenariu/caz de utilizare este distribuit între obiectele participante
  - În etapa de analiză, permit identificarea unor descrieri de comportament ambigue sau a unor obiecte participante lipsă
  - Datorită notaţiei utilizate, nu sunt, în general, o metodă de comunicare cu clienţii/utilizatorii la fel de eficientă ca şi scenariile/cazurile de utilizare
- Euristici de realizare a diagramelor de secvenţă de analiză
  - Prima coloană trebuie să corespundă actorului care a iniţiat cazul de utilizare
  - A doua coloană trebuie să corespundă obiectului *boundary* folosit de actor pentru a iniţia cazul de utilizare
  - A treia coloană trebuie să corespundă obiectului *control* care gestionează cazul de utilizare
  - Obiectele *control* sunt create de către obiectele *boundary* care iniţiază cazurile de utilizare
  - Ulterior, obiectele *boundary* sunt create de obiecte *control*
  - Obiectele *entity* sunt accesate de către obiectele *boundary* şi *control*, dar nu accesează niciodată obiecte *boundary* sau *control*

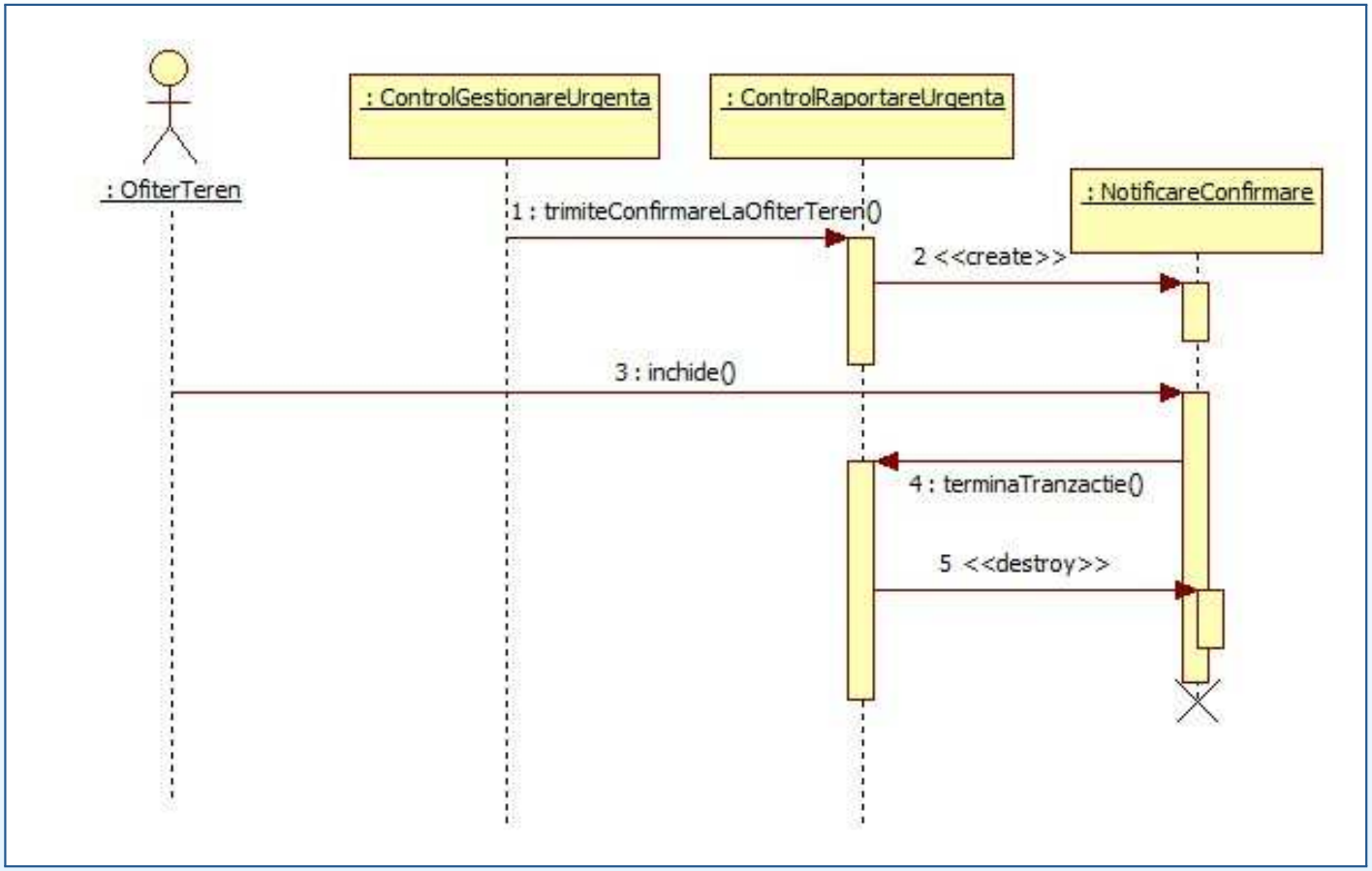
CU *Raportează Urgenţă* - diagramă de secvenţă



CU *Raportează Urgenţă* - diagramă de secvenţă (cont.)



CU *Raportează Urgenţă* - diagramă de secvenţă (cont.)



CU *Raportează Urgenţă* - rafinare

- Clasă *entity* nouă

Nume clasă	**Descriere**
Confirmare	Răspuns al unui *Dispecer* la un *RaportUrgenţă* trimis de către un *OfiţerTeren*. Trimiţând o *Confirmare*, *Dispecer*ul îi comunică ofiţerului că a primit raportul, a creat un *Eveniment* şi i-a asignat resurse. *Confirmare*a include specificarea resurselor alocate şi timpul estimat al sosirii lor.

- Rafinarea descrierii cazului de utilizare

Nume	*Raportează Urgenţă*
...	
Flux de evenimente	... 5. ... *Confirmare*a îi indică ofiţerului că raportul a fost primit, evenimentul a fost creat şi resursele au fost alocate. *Confirmare*a include specificarea resurselor (ex. camion pompieri) şi momentul estimativ al sosirii lor. ...
...	

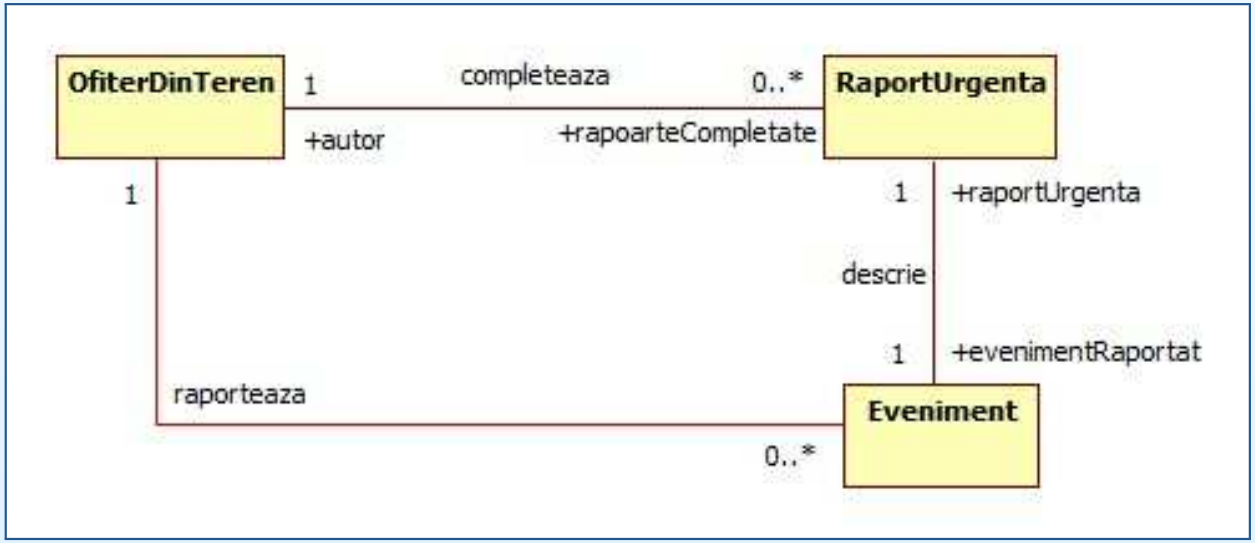
Identificarea asocierilor

- O *asociere* indică o relaţie între două sau mai multe clase
  - Ex.: Un ofiţer din teren scrie rapoarte de urgenţă.

  - Proprietăţi ale unei asocieri
    - nume
    - roluri
    - multiplicităţi
  - Avantaje ale modelării asocierilor în analiză
    - Clarificarea modelului de analiză prin explicitarea modului de relaţionare al obiectelor
      - Raportul este scris de un ofiţer şi nu de un dispecer.
    - Investigarea cazurilor limită
      - Există rapoarte cu mai mulţi autori? Dar anonime?

Identificarea asocierilor (cont.)

- Euristici pentru reprezentarea asocierilor
  - Examinarea structurilor verbale
  - Numirea explicită a asocierilor şi rolurilor
  - Utilizarea calificatorilor pentru a identifica spaţii de nume şi atribute cheie
  - Eliminiarea tuturor asocierilor redundante (derivabile din alte asocieri)
  - Reprezentarea multiplicităţilor doar după stabilizarea mulţimii de asocieri
  - Un număr exagerat de asocieri generează redundanţă şi afectează negativ inteligibilitatea modelului
- Model cu asocieri redundante

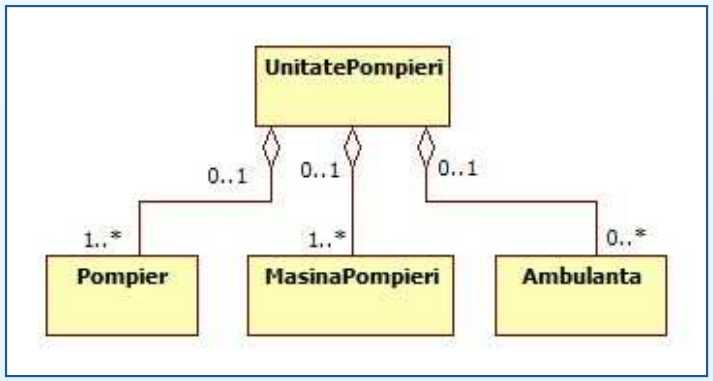


Identificarea agregărilor

- *Agregare*a reprezintă un tip particular de asociere, ce denotă o relaţie de tip parte-întreg
- *Compunerea* este un tip particular de agregare, în care existenţa părţilor e condiţionată de întreg

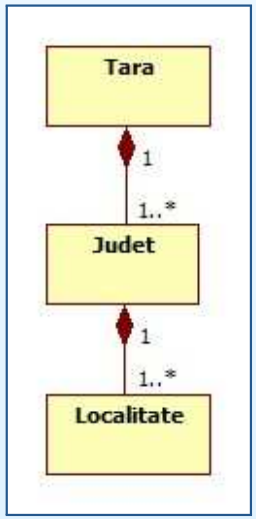
Ex.: Agregare

- Un pompier face parte din cel mult o unitate la un moment dat, însă poate fi reasignat la o altă unitate pe parcursul existenţei sale



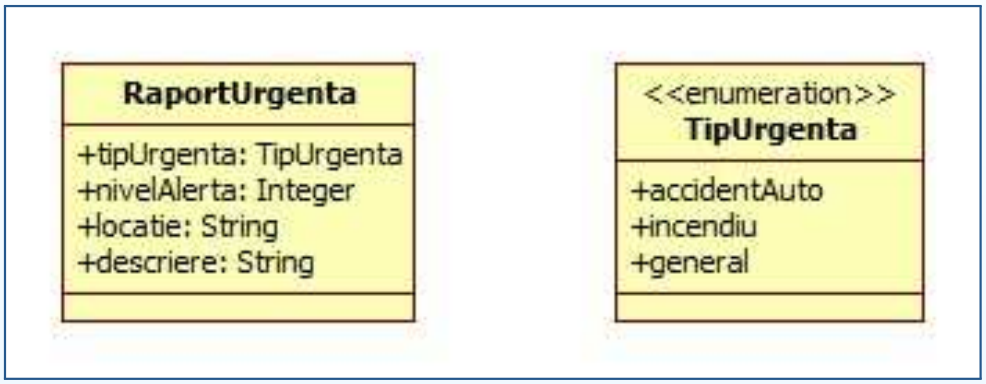
Ex.: Compunere

- O localitate e parte a unui singur judet
- O aceeaşi localitate nu va putea fi parte a unui alt judeţ, nici partajată cu alte judeţe



Identificarea atributelor

- *Atributele* reprezintă proprietăţi ale obiectelor individuale

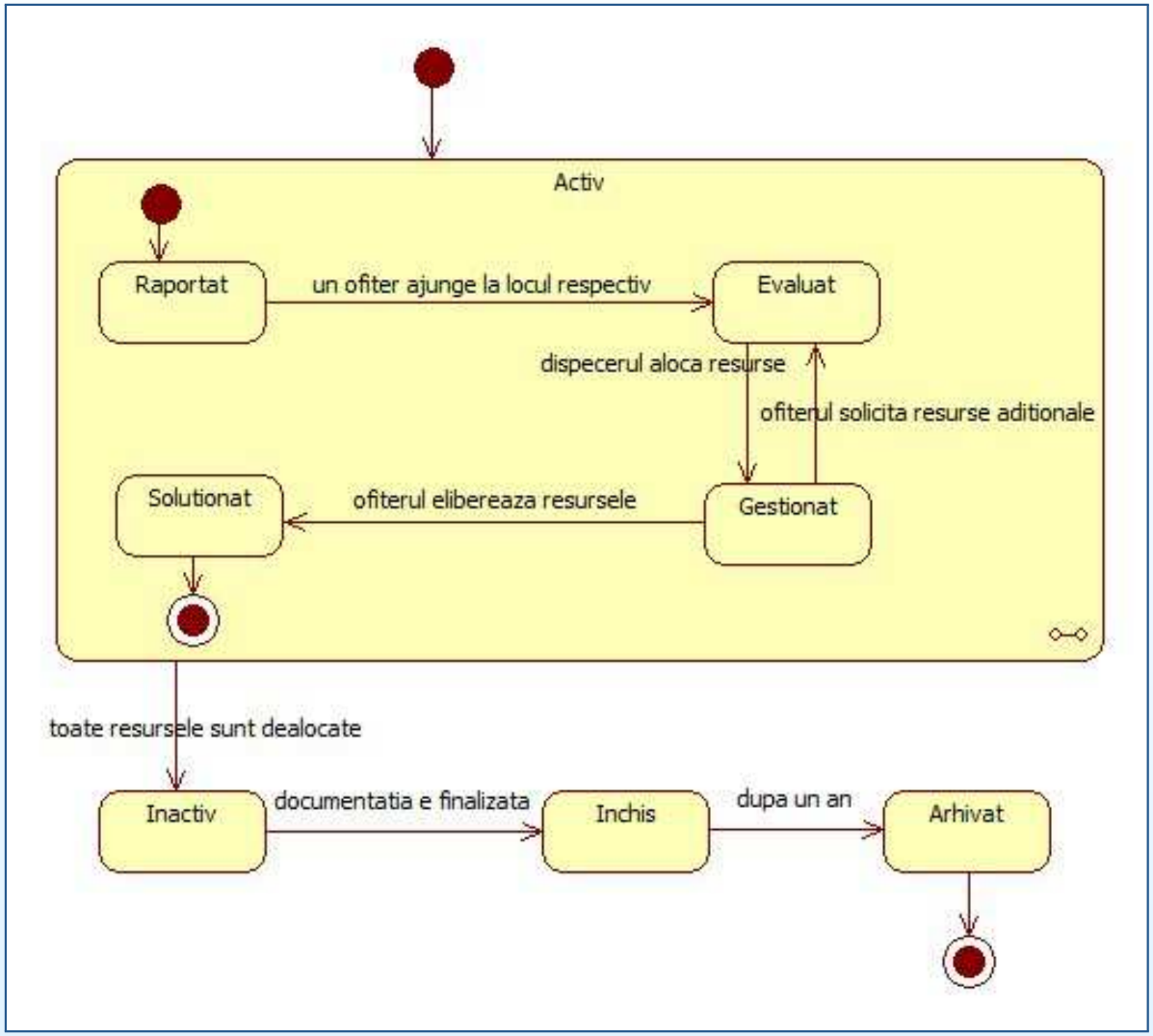


- Euristici pentru identificarea atributelor
  - Examinarea construcţiilor substantivale posesive (ex.: descrierea urgenţei)
  - Reprezentarea proprietăţilor persistente ca şi atribute
  - Identificarea doar a acelor atribute relevante pentru sistemul în cauză
  - Amânarea identificării atributelor mai puţin relevante din punct de vedere al funcţionalităţii până după stabilizarea modelului obiectual
  - Descrierea sumară a fiecărui atribut
  - Proprietăţile de tip obiect nu se reprezintă ca şi atribute, în acest caz se folosesc asocieri!

Modelarea comportamentului cu diagrame de stări

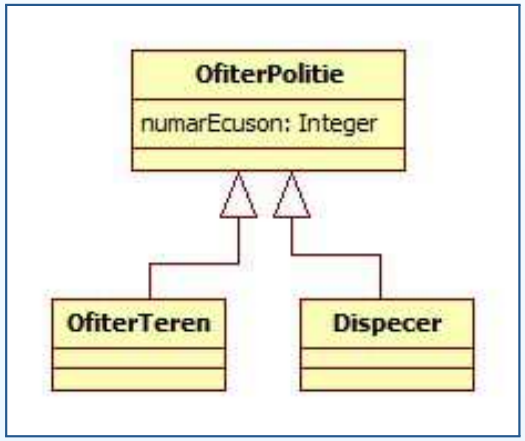
- Modelarea comportamentului
  - Diagrame de secvenţă - prezintă comportamentul din perspectiva unui singur caz de utilizare
  - Diagrame de tranziţie a stărilor - prezintă comportamentul din perspectiva unui singur obiect
- Rolul diagramelor de tranziţie a stărilor în etapa de analiză constă în identificarea unor cazuri de utilizare omise sau a unor omisiuni în descrierea cazurilor de utilizare existente
- Nu este necesară construirea unei diagrame de tranziţie a stărilor pentru fiecare clasă a modelului obiectual, doar pentru cele ale căror obiecte au durata de viaţă mare şi comportament complex, dependent de stare
  - Cel mai frecvent pentru obiecte *control*, mai puţin frecvent pentru obiecte *entity* şi aproape niciodată pentru obiecte *boundary*

CU *Raportează Urgenţă* - diagrama de stări pentru *Eveniment*

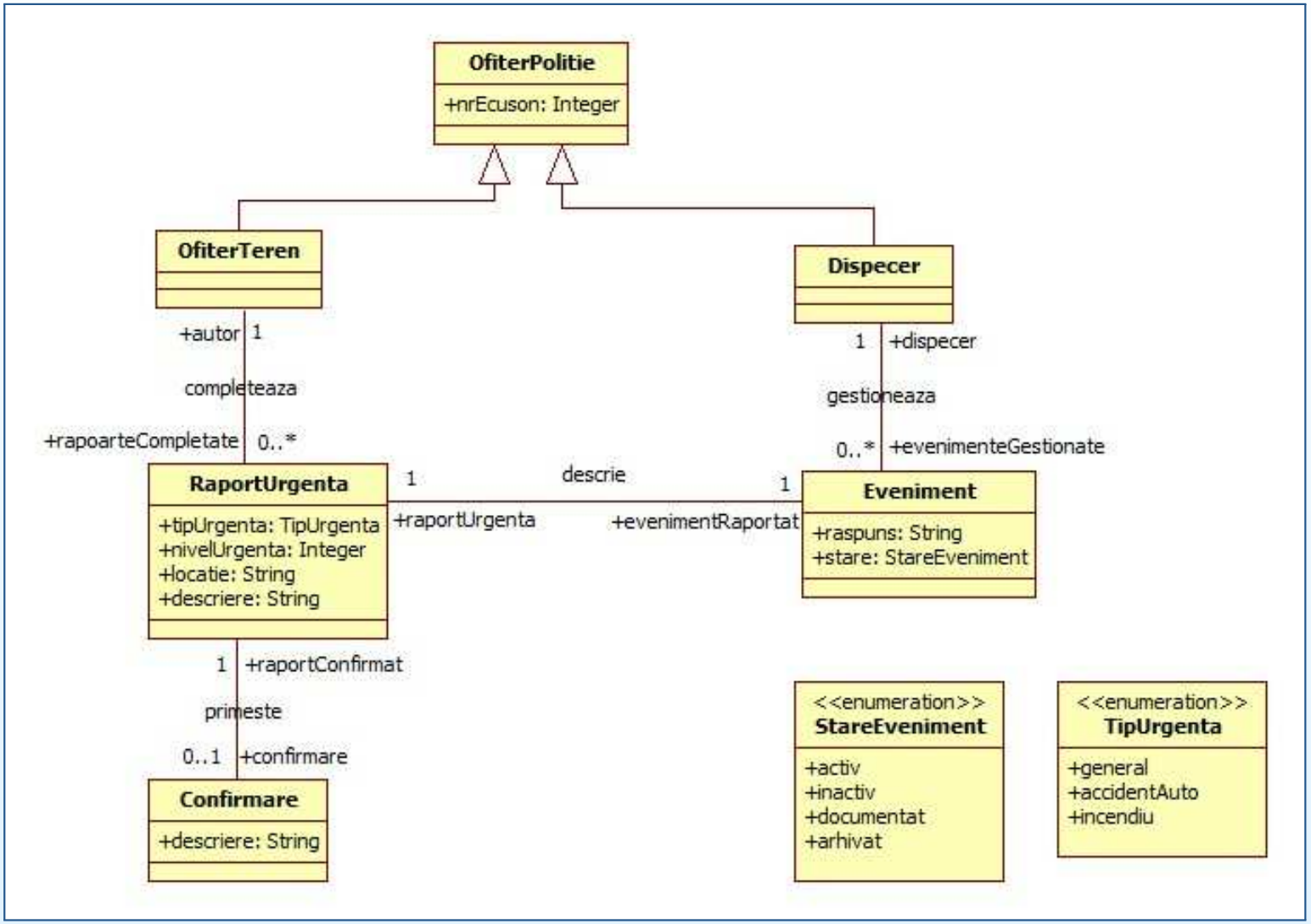


Identificarea ierarhiilor de clase

- Rolul generalizărilor îl constituie eliminarea redundanţelor din modelul obiectual
- Ex.:



CU *Raportează Urgenţă* - model al claselor *entity* (conceptual)



Revizuirea modelului de analiză

- Modelul de analiză se dezvoltă iterativ-incremental
- Odata ce devine stabil, urmează revizuirea acestuia
  - Revizuire internă (dezvoltatori) + revizuire dezvoltatori şi client
  - E specificarea cerinţelor corectă, completă, consistentă şi neambiguă? Sunt cerinţele realiste şi verificabile?
  - Revizuirea e facilitată de existenţa unor liste de întrebări posibile
- Asigurarea *corectitudinii*
  - Este dicţionarul claselor *entity* înţeles de către client?
  - Corespund clasele abstracte unor concepte utilizator?
  - Sunt toate descrierile conforme cu definiţiile date de utilizatori?
  - Au toate obiectele *entity* şi *boundary* ca şi nume construcţii substantivale sugestive?
  - Au toate cazurile de utilizare şi obiectele *control* ca şi nume construţii verbale sugestive?
  - Sunt toate cazurile de eroare descrise şi gestionate?

Revizuirea modelului de analiză (cont.)

- Asigurarea *completitudinii*
  - Pentru fiecare obiect: Este folosit de vreun caz de utilizare? În cadrul cărui caz de utilizare este creat? modificat? distrus? Poate fi accesat pornind de la un obiect *boundary*?
  - Pentru fiecare atribut: Unde ii este atribuită o valoare? Care este tipul său? Poate reprezenta un calificator?
  - Pentru fiecare asociere: Când este traversată? De ce a fost aleasă respectiva multiplicitate?
  - Pot fi calificate asocierile *one-to-many* şi *many-to-many*?
  - Pentru fiecare obiect *control*: Are asocierile necesare accesării tuturor obiectelor din respectivul caz de utilizare?
- Asigurarea *consistenţei*
  - Există mai multe clase sau cazuri de utilizare cu acelaşi nume?
  - Denotă entităţile cu nume similare concepte înrudite?
  - Există obiecte cu atribute şi asocieri similare ce nu fac parte din aceeaşi ierarhie de moştenire?

Revizuirea modelului de analiză (cont.)

- Asigurarea *realismului*
  - Pot fi îndeplinite cerinţele legate de performanţă şi fiabilitate?
  - Au fost aceste cerinţe verificate folosind prototipuri pe platforma hardware aleasă?

Şablonul documentului de analiză a cerinţelor

- 1. Introducere
  - 1.1 Scopul sistemului
  - 1.2 Obiectivele şi criteriile de succes ale proiectului
  - 1.3 Definiţii, acronime şi abrevieri
  - 1.4 Referinţe
  - 1.5 Sumar
- 2. Sistemul curent
- 3. Sistemul propus
  - 3.1 Sumar
  - 3.2 Cerinţe funcţionale
  - 3.3 Cerinţe nefuncţionale
    - 3.3.1 Utilizabilitate
    - 3.3.2 Fiabilitate
    - 3.3.3 Perfomanţă
    - 3.3.4 Suportabilitate
    - 3.3.5 Implementare
    - 3.3.6 Interfaţă

Şablonul documentului de analiză a cerinţelor (cont.)

- 3.3.7 Instalare
- 3.3.8 Cerinţe legale

- 3.4 Modele

- 3.4.1 Scenarii
- 3.4.2 Modelul cazurilor de utilizare
- 3.4.3 Modelul obiectual
- 3.4.4 Modelul dinamic
- 3.4.5 Prototipul interfeţei utilizator

- 4. Glosar

Referinţe

- [Abbott, 1983] R. Abbott, *Program design by informal English descriptions*, Communications of the ACM, Vol. 26, No. 11, 1983.